ลออ เพิ่มสมบัติ

การคอมพายล์แยกส่วน ในเทอร์โบซี

การคอมพายล์แยกส่วนสามารถช่วยในการเขียนโปรแกรมโครงสร้างที่ดี ที่เรียกว่า โปรแกรมโมดูล

ตามปกติเมื่อเขียนโปรแกรมแล้ว เราจะคอมพายลิงค์โปรแกรมนั้นครั้งเดียว อันที่จริงแล้วโปรแกรมหนึ่งมีหลายฟังก์ชัน แต่เราทำให้เป็นหน่วยเดียวเพื่อสะดวกในการคอมพายล์ อย่างไรก็ตาม เราสามารถจะตัดทอนโปรแกรมนั้นออกเป็นส่วน ๆ เอาแต่ละส่วนไปเก็บไว้ในไฟล์หลาย ๆ ไฟล์ แต่ละไฟล์อาจจะมีฟังก์ชันเดียวหรือมากกว่า แล้วเอาแต่ละไฟล์นี้มาคอมพายล์ทีละไฟล์ แล้วค่อยลิงค์เข้ากันเป็นไฟล์นามสกุล .exe ที่จะนำไปรันได้ ดูตัวอย่างที่ 1

```
/* mainprog.c */
/* main program to test sepatate
Compilation */
int formula (int, int); /* prototype/
void main (void)
 {
   int a, b, ans ;
   printf (" Type two integers : ");
   scanf ("% d % d", &a, &b);
   ans = formula (a,b);
   printf ("The sum of the squares is
% d.", ans);
}
```

ตัวอย่างที่ 1

ลองพิมพ์โปรแกรม mainprog.c โดยใช้เอดิเตอร์ของเทอร์โบซี แล้วทำสำเนาไว้โดยการกด [F2] คุณก็จะมีไฟล์ชื่อ mainprog.c ยังไม่ต้องคอมพายล์ ถ้าคุณคอมพายล์จะมีข้อความผิดพลาด (error) เมื่อคอมพายเลอร์พบฟังก์ชัน formula ():

```
ans = formula (a,b);
```

ใช้วิธีเดียวกับข้างต้น พิมพ์โปรแกรม formula.c แล้วทำสำเนาเก็บไว้ (ดูตัวอย่างที่ 2)

```
/* formula.c */
/* function returns sum of squares of
arguments */
int formula (int x, int y)
{
return (x*x + y*y);
}
```

ตัวอย่างที่ 2

จะได้ไฟล์ที่เรียกว่า formula.c ซึ่งก็จะไม่มีฟังก์ชัน formula () ในไฟล์ข้างต้น

เราลองดูซิว่าจะรวมไฟล์ 2 ไฟล์นี้ เป็นโปรแกรมที่สมบูรณ์ได้อย่างไร

การใช้โปรเจ็กต์ (project) ในเทอร์โบซี

คอมพายเลอร์ในภาษาซีจะมีวิธีการรวมไฟล์หลาย ๆ ไฟล์เข้าด้วยกัน เป็นไฟล์ที่มีนามสกุล .exe อยู่ 2 วิธี คือ

1. วิธีทำด้วยมือ คือพิมพ์คำสั่งเป็นแบบทีละบรรทัด คอมพายล์ทีละส่วนแล้วลิงค์เข้าด้วยกัน

2. ใช้ยูทิลิตี้ Make (เป็นโปรแกรมที่สร้างขึ้นมาเพื่อจะคอมพายล์โดยอัตโนมัติ)

วิธีการของ Make ช่วยให้การคอมพายล์โปรแกรมใหญ่ ๆ ที่ประกอบด้วยไฟล์หลายไฟล์ อาจจะมากกว่า 12 ไฟล์ขึ้นไป เมื่อต้องการแก้ไขบางไฟล์ก็คอมพายล์เฉพาะไฟล์ที่แก้ไข ไม่จำเป็นต้องคอมพายล์ใหม่ทั้งหมด (สำหรับไฟล์อื่นที่ถูกต้องอยู่แล้ว)

การสร้างโปรเจ็กต์ไฟล์ (ไฟล์ที่มีนามสกุล .PRJ)

หัวใจของกรรมวิธีของโปรเจ็กต์คือ ไฟล์ที่มีนามสกุล .PRJ ไฟล์นี้จะประกอบด้วยชื่อของทุกไฟล์ที่ต้องการจะรวมกันเป็นโปรแกรมเดียว

วิธีการเชื่อมไฟล์เข้าด้วยกัน เริ่มแรกต้องสร้างไฟล์ต้นฉบับที่มีนามสกุล .c (หรือไฟล์ที่คอมพายล์เป็น .lib หรือ .obj แล้ว) และต้องมั่นใจว่าไฟล์เหล่านี้อยู่ในไดเรคทอรีปัจจุบันเรียบร้อยแล้ว ขั้นต่อไปก็เรียกเทอร์โบซีแล้วเลือกเปิด Open Project จากเมนูโปรเจ็กต์พิมพ์ชื่อไฟล์ลงไปในช่องโหลดโปรเจ็กต์ไฟล์ แล้วต่อท้ายด้วยนามสกุล .PRJ เช่น sumsqr.prj (ชื่อที่ใช้จะเปลี่ยนเป็น sumsqr.exe ในภายหลัง) กด [ENTER] หลังจากที่พิมพ์ชื่อไฟล์แล้ว หลังจากนั้นไฟล์ Load Project จะหายไปที่หน้าต่างจะขึ้นไฟล์โปรเจ็กต์ SUMSQR (หรือชื่ออื่นที่คุณพิมพ์เข้าไป) และจะมีที่ว่างตอนล่างของจอภาพ พร้อมกับมีคำสั่งให้เพิ่มไฟล์เข้าไป คุณก็กด [insert] จะขึ้นข้อความในกรอบว่า "Add Item to Project List" และจะมีไฟล์ที่มีนามสกุล .c ในไดเรคทอรีปัจจุบัน การเพิ่มไฟล์เข้าไปก็เพียงลากเม้าส์แล้วคลิ๊กที่ปุ่ม Add (หรือจะใช้ลูกศร ลากไปที่ Add แล้วกด Enter)

สำหรับตัวอย่างนี้คุณก็เลือก mainprog.c และ formula.c เมื่อเลือกแล้วแต่ละโปรแกรมก็จะเข้าไปอยู่ในโปรเจ็กต์ คุณก็ดูจากรายการของโปรเจ็กต์ที่อยู่ใน Project window ถ้าต้องการเพิ่มไฟล์ .LIB หรือ .OBJ ก็พิมพ์เข้าไปในฟิลด์ NAME

เมื่อรายการในโปรเจ็กต์ครบถ้วนแล้วก็กด [F10] เพื่อกลับไปเมนูหลัก (main menu) เลือก Make จากเมนูคอมพายล์หรือกด [F9] คุณก็จะพบว่าเทอร์โบซีคอมพายล์ไฟล์ต่าง ๆ ในที่นี้ mainproj.c และ formula.c แล้วก็ลิงค์ .obj เป็น sumsqr.exe หลังจากนั้นก็รันโปรแกรมโดยเลือกรันจากเมนูหลัก หรือออกจากเทอร์โบซีไปที่ดอสแล้วพิมพ์ SUMSQR ก็จะปรากฏผลดังนี้

```
Type two integers : 4 3
The sum of the Squares is 25
```

ต่อไปถ้าคุณต้องการเปลี่ยนแปลงไฟล์ใดไฟล์หนึ่ง การสร้างโปรเจ็กต์ทั้งหมดขึ้นมาใหม่ก็จะทำได้รวดเร็วขึ้น โดยการเปิดไฟล์โปรเจ็กต์จาก Make แล้วเลือกไฟล์ .PRJ ที่สร้างขึ้นได้จากไฟล์บ็อกซ์ หน้าต่างโปรเจ็กต์ที่มีไฟล์ต้นฉบับจะปรากฏให้เห็น หลังจากนั้นเลือก Make จากเมนูคอมพายล์เทอร์โบซีจะสร้างโปรเจ็กต์ให้คุณเอง และจะคอมพายล์เฉพาะไฟล์ที่คุณแก้ไขใหม่

ประโยชน์ของการคอมพายล์แยกส่วน

1. โปรแกรมเมอร์จะเขียนเฉพาะส่วนที่ตัวเองรับผิดชอบก็คอมพายล์เฉพาะของตนเอง ถ้าคอมพายล์ผ่านแล้วเวลามารวมกันกับไฟล์อื่นก็ไม่ต้องคอมพายล์ใหม่ทั้งหมด
2. ช่วยให้การเขียนโปรแกรมเป็นแบบโครงสร้างที่ดี
3. สามารถจะใช้ไฟล์ในไลบรารีได้

วิธีที่ไฟล์แต่ละไฟล์แลกเปลี่ยนข้อมูล

จากตัวอย่างของเรา mainprog () และ formula () ถูกลิงค์เข้าด้วยกัน มีการส่งเลข 2 จำนวน จาก mainprog () ไปยัง formula () ด้วยการใช้อาร์กิวเมนต์ (argument) แล้ว formula () จะส่งผลลัพธ์ที่คำนวณได้กลับไปยัง mainprog () ด้วยคำสั่ง return วิธีเช่นนี้ไม่จำเป็นต้องมีประโยคพิเศษในแต่ละฟังก์ชัน แต่ถ้าเราต้องการให้ฟังก์ชันอื่น ๆ ที่แยกคอมพายล์ได้มีโอกาสใช้ตัวแปรนี้ด้วยจะต้องมีประโยคพิเศษเพิ่มเข้ามาคือต้องมี extern นำหน้าตัวแปรนั้น ๆ เพื่อจะให้เห็นวิธีการ ดูตัวอย่างที่ 3

```
/* mainprog.c */
/* main program to test reporate compilation */
int formula (void) ; /* prototype */
int a, b ; /* definition of external variables */
void main (void)
{
int ans ;
printf ("Type two integers :") ;
scanf ("%d %d, &a, &b);
ans = formula () ;
printf ("The sum of the square is %d", ans) ;
}
```

ตัวอย่างที่ 3

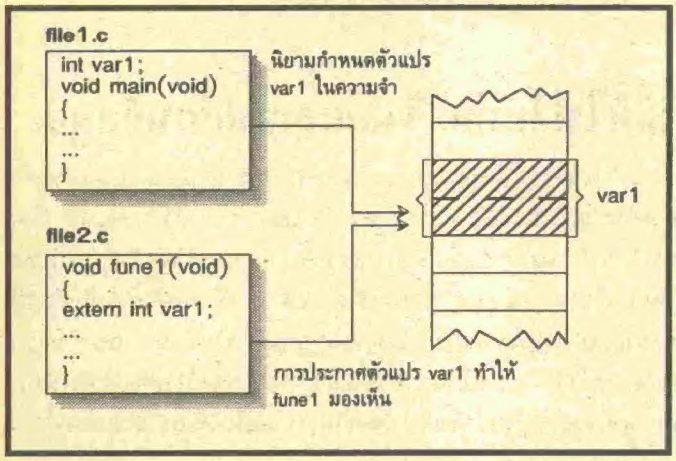
โปรแกรมนี้เหมือนกับ mainprog.c ที่เขียนขึ้นมาก่อน ยกเว้นที่ย้ายเอาตัวแปร a, b ที่เก็บค่าของจำนวนเต็ม (ที่ผู้ใช้ป้อนเข้ามาไปไว้นอกฟังก์ชัน) ซึ่งต่อไปจะเรียกว่าตัวแปรภายนอก (external) ต่อไปถ้า mainprog.c เรียก formula () ก็ไม่จำเป็นต้องส่งผ่านอาร์กิวเมนต์อีกแล้ว ดูตัวอย่างที่ 4

```
/* formula 2.c */
/* function returns sum of squares of arguments
*/
/* uses global variables for input */
int formula (void)
{
extern int a, b ; /* declaration of external
variables */
return (a*a + b*b) ;
}
```

ตัวอย่างที่ 4

จะเห็นว่าไม่จำเป็นต้องประกาศตัวแปรใด ๆ ให้เป็นอาร์กิวเมนต์ฟอร์มอล (formal) แก่ฟังก์ชันอีกแล้ว แต่ประกาศตัวแปร a, b เป็น external int ซึ่งหมายความว่า extern เป็นตัวอย่างของ storage class ซึ่งจะกล่าวรายละเอียดในบทความต่อไป แต่ในที่นี้เพียงให้รู้ว่า extern ที่อยู่หน้าต่างตัวแปร ไฟล์อื่นจะสามารถใช้ตัวแปรนี้ได้ extern เป็นคีย์เวิร์ดที่บอกให้คอมพายเลอร์ทราบว่า ได้นิยามตัวแปรนี้ไว้แล้วในที่ใดที่หนึ่ง

ในไฟล์อื่น เมื่อลิงค์เกอร์พบข้อความนี้ที่คอมพายเลอร์ส่งเข้ามาก็จะค้นหาในไฟล์อื่น เมื่อพบแล้วก็ทำการเชื่อมเข้าด้วยกันดังรูปที่ 1



รูปที่ 1

ไฟล์ไลบรารี

เราได้เห็นแล้วถึงวิธีการเอาฟังก์ชันมาใช้ในโปรแกรม อาจจะรวมกันในโปรแกรมเดียวหรือแยกออกเป็นไฟล์แล้วคอมพายล์แยกส่วนแล้วลิงค์เข้าด้วยกันอีกทีหนึ่ง ยังมีอีกวิธีหนึ่งที่เป็นไปได้คือ นำเอาฟังก์ชันที่อยู่ในไฟล์ไลบรารี (library) ไลบรารีคือ ไฟล์ที่ประกอบขึ้นด้วยฟังก์ชัน ฟังก์ชันจะถูกรวมกันให้เป็นกลุ่มที่เรียกว่า โมดูล (Module) ซึ่งแต่ละโมดูลมีลักษณะเฉพาะ เมื่อคุณลิงค์โปรแกรมด้วยไลบรารีมันก็จะรวมเข้ามาอยู่ในไฟล์นามสกุล .exe

โปรแกรมเมอร์ภาษาซีส่วนใหญ่จะซื้อไลบรารีพิเศษมาเฉพาะที่ไม่มีในไลบรารีซีมาตรฐาน อาทิ การสร้างเมนู ฟังก์ชันทางสถิติ ฐานข้อมูล และอื่น ๆ หรืออาจจะพัฒนาขึ้นมาเอง

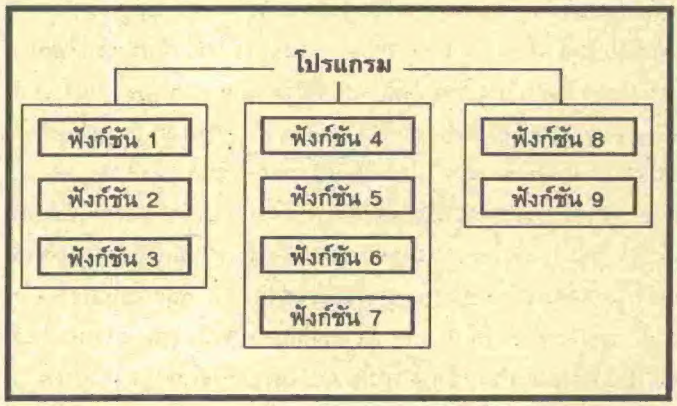
การโปรแกรมโมดูล และการคอมพายล์แยกส่วน

เหตุผลสำคัญที่มีการคอมพายล์แยกส่วน เพื่อจะช่วยในการเขียนโปรแกรมโครงสร้างที่ดีที่เรียกว่า โปรแกรมโมดูล

คุณคงทราบแล้วว่าการเขียนโปรแกรมที่ดีไม่ควรเขียนโปรแกรมขนาดใหญ่เพียงหน่วยเดียว ทางที่ดีควรจะแบ่งโปรแกรมขนาดใหญ่ให้เป็นโปรแกรมย่อย ๆ ซึ่งจะช่วยให้ทำการแก้ไขได้ง่าย และทำความเข้าใจแต่ละส่วนได้ง่ายกว่า โปรแกรมในภาษาซีจะต้องประกอบด้วยฟังก์ชัน main () ซึ่งจะเรียกใช้ฟังก์ชันอื่น ๆ เมื่อต้องการใช้งานฟังก์ชันนั้น และฟังก์ชันอื่น ๆ ก็จะสามารถเรียกใช้อีกฟังก์ชันหนึ่งได้

แต่ละฟังก์ชันควรจะทำงานเฉพาะอย่าง เช่น คำนวณเงินได้ ฟังก์ชันนี้อาจเรียกใช้ฟังก์ชันคำนวณภาษี ฟังก์ชันคำนวณภาษีอาจเรียกใช้ฟังก์ชันเพื่อดูอัตราภาษีจากตาราง จุดสำคัญประการแรกของฟังก์ชันจะต้องนิยามไว้อย่างดี ฟังก์ชันจะต้องไม่ใหญ่เกินไป กฎทั่ว ๆ ไป ฟังก์ชันหนึ่ง ๆ ไม่ควรยาวเกิน 1 หน้ากระดาษ

โปรแกรมใหญ่ ๆ ควรแยกออกเป็นไฟล์ แต่ละไฟล์แยกออกเป็นฟังก์ชันดังรูปที่ 2



รูปที่ 2

ออกแบบโปรแกรม

ซียังขาดเครื่องมือสำหรับออกแบบโปรแกรมโครงสร้างที่ดี แต่ก็ขึ้นอยู่กับโปรแกรมเมอร์ว่าจะใช้มันได้อย่างมีประสิทธิภาพหรือไม่

การเขียนโปรแกรมใหญ่ ๆ ควรจะวางแผนก่อนที่จะลงมือเขียน แต่ละบรรทัด กำหนดฟังก์ชันขึ้นมาก่อนว่าแต่ละฟังก์ชันมีจุดมุ่งหมายอะไร จะส่งอะไรผ่านอาร์กิวเมนต์ในฟังก์ชัน และฟังก์ชันจะคืนผลลัพธ์อะไรมาให้แก่ฟังก์ชันที่เรียกใช้ โปรแกรมเมอร์ส่วนใหญ่จะใช้วิธีจากบนลงล่าง (top down) คือ เริ่มต้นด้วยฟังก์ชัน main () แล้วไล่ลงไปหาฟังก์ชันที่ทำงานระดับต่ำลงมา เช่น การนำเอาอักษรออกจอภาพ บางสถานการณ์ก็ต้องพิจารณาฟังก์ชันระดับล่างควบคู่ไปกับฟังก์ชันที่อยู่เหนือกว่า